Attorney Docket No. 10559-1010

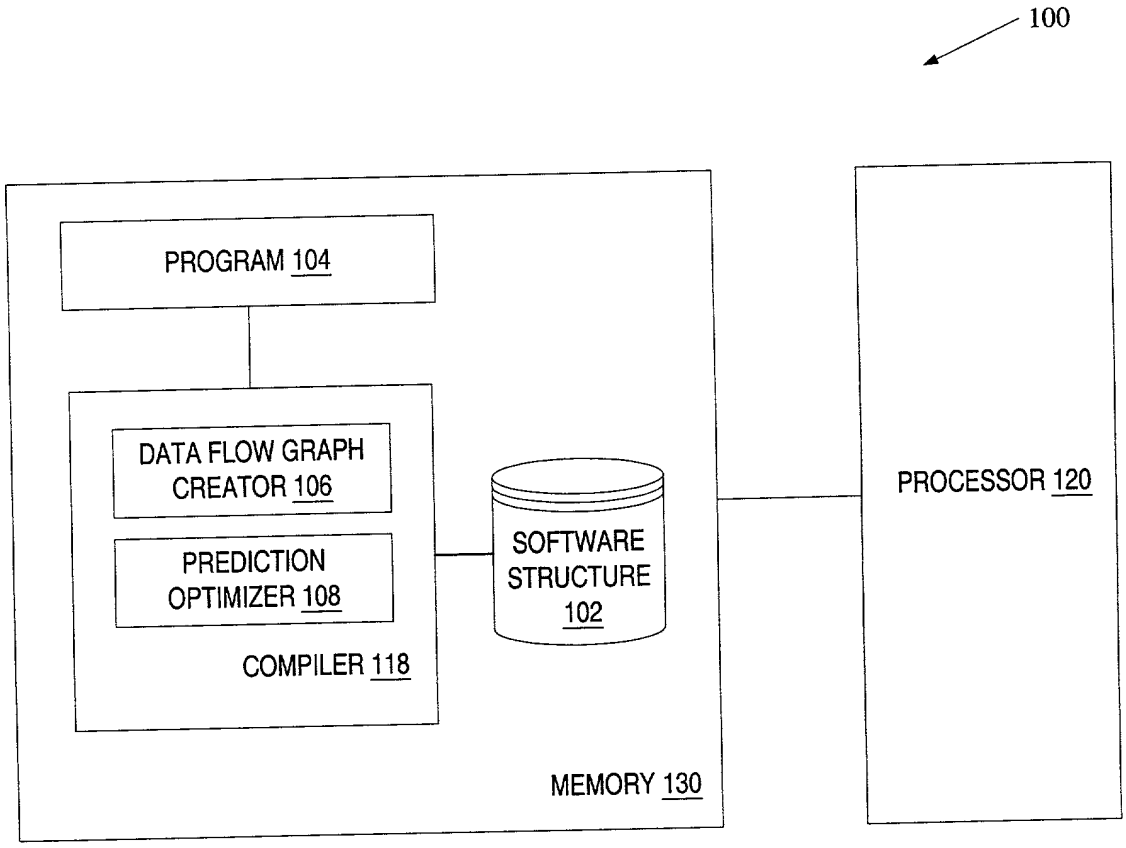


FIG. 1

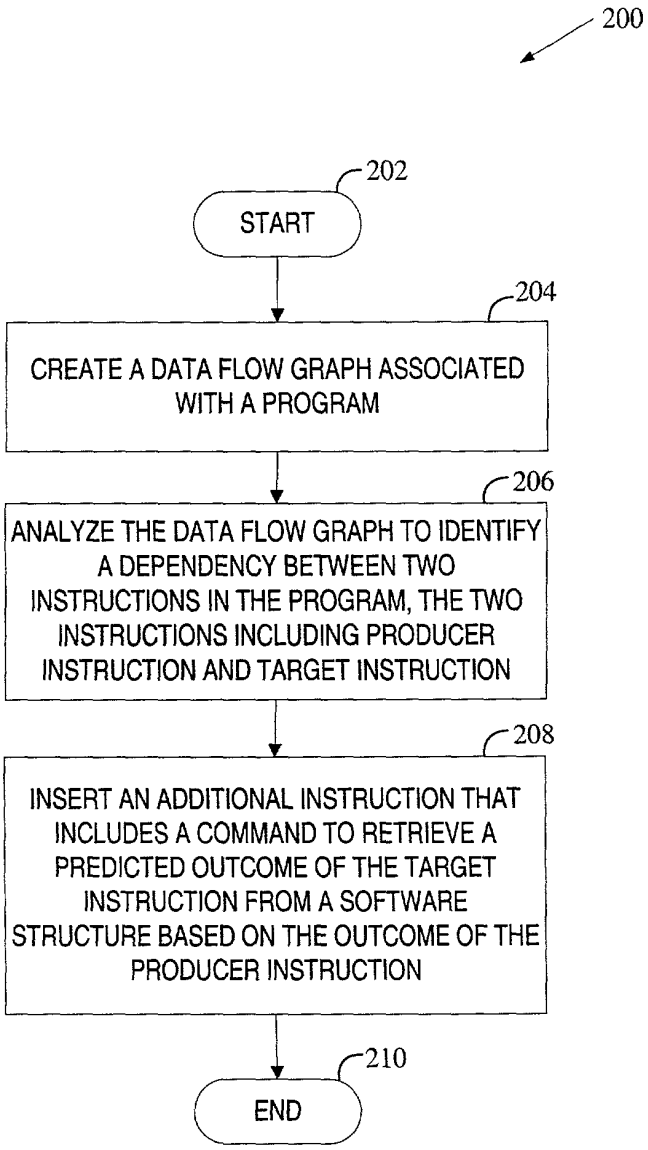
200
202
START
204
CREATE A DATA FLOW GRAPH ASSOCIATED WITH A PROGRAM
206
ANALYZE THE DATA FLOW GRAPH TO IDENTIFY A DEPENDENCY BETWEEN TWO INSTRUCTIONS IN THE PROGRAM, THE TWO INSTRUCTIONS INCLUDING PRODUCER INSTRUCTION AND TARGET INSTRUCTION
208
INSERT AN ADDITIONAL INSTRUCTION THAT INCLUDES A COMMAND TO RETRIEVE A PREDICTED OUTCOME OF THE TARGET INSTRUCTION FROM A SOFTWARE STRUCTURE BASED ON THE OUTCOME OF THE PRODUCER INSTRUCTION
210
END

FIG. 2

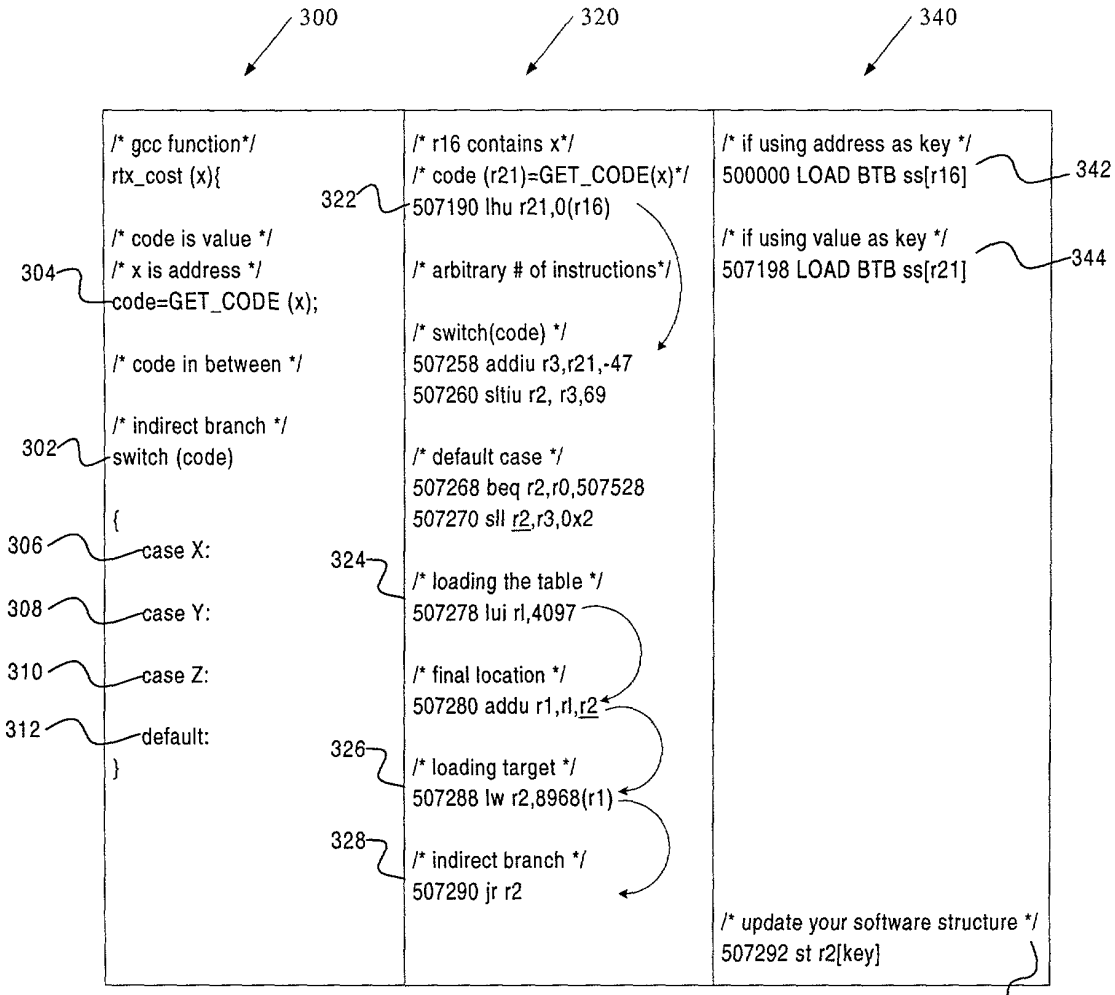


FIG. 3A FIG. 3B FIG. 3C

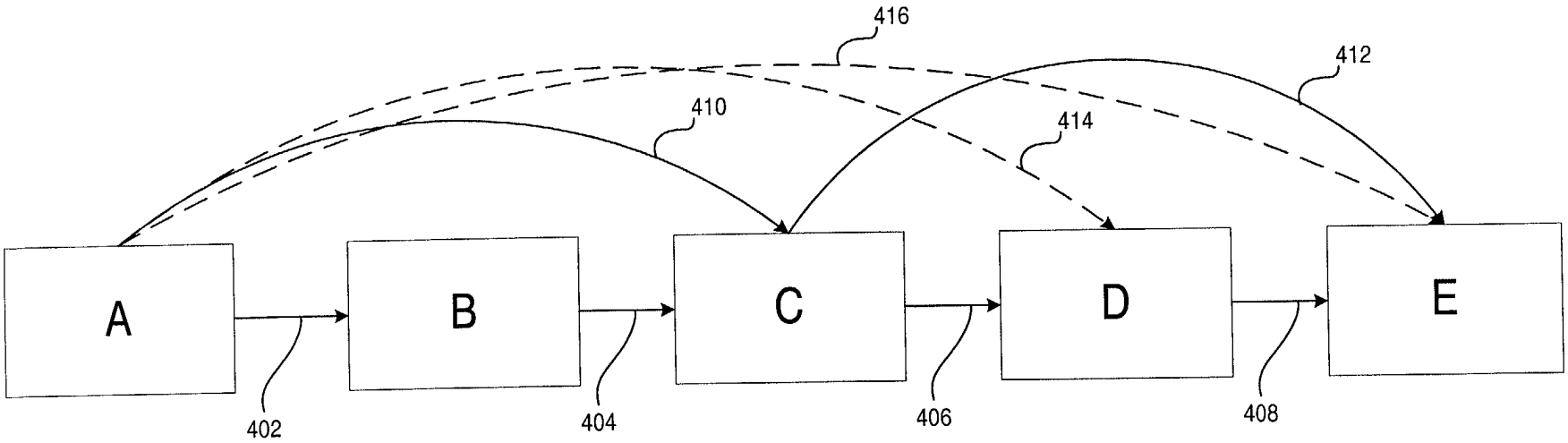


FIG. 4

ptr1

```
/* frequency execute "go" */
/* not fully complete here */
/* merge list1 into list2 */
/* each loop iteration is
dependent on the previous
iteration */

mrglist(int ptr1, int ptr2){

/* traversing the linked list */
 while (ptr1 !=EOL){
  /* use ptr1 */
  ptr1=links[ptr1];
 }
}
```

FIG. 5A

```
mrglist:

loop:

 if(ptr1 == EOL) goto end;

/* use ptr1 */
 ptr1 = links[ptr1];

 goto loop;
end:
```

FIG. 5B

```
/* modernized pseudo-code */
/* update is not in code.... */
/* assume perfect bp */
mrglist:
loop:
 if (ptr1 == EOL) goto end;
 /* use ptr1 */
 /* MLP */
 nextptr1 = links[ptr1];
 specptr1 = ss[ptr1];

 ptr1 = nextptr1;
 if (ptr1 == EOL) goto end;
 /* use ptr1 */
 nextptr1 = links[ptr1];
 if(nextptr1 == specptr1){
  /* speculatively correct */
  ptr1 = specptr1;
  }else{
   /* speculatively incorrect */
   ptr1 = nextptr1;
  }
 goto loop;
end;
```

502, 504, 506, 508

FIG. 5C